07-03-00

A

EXPRESS MAIL LABEL NO. EM598711332US

Attorney Docket No. 72255/02662

Box Patent Application
Assistant Commissioner for Patents
Washington, DC 20231

NEW APPLICATION TRANSMITTAL

Transmitted herewith for filing is the patent application of

Inventor(s) : Kenneth W. Batcher

For (title) : REPEAT FUNCTION FOR PROCESSING OF REPETITIVE INSTRUCTION STREAMS

1. Type of Application

This new application is for a(n):

(X) Original (nonprovisional)
( ) Continuation
( ) Continuation-in-part (CIP)
( ) Divisional
( ) Design
( ) Plant

NOTE: If continuation, CIP or divisional, then complete section 2.

CERTIFICATION UNDER 37 C.F.R. 1.10*
(Express Mail label number is mandatory.)
(Express Mail certification is optional.)

I hereby certify that this New Application Transmittal and the documents referred to as attached therein are being deposited with the United States Postal Service on this date June 30, 2000, in an envelope as "Express Mail Post Office to Addressee," mailing Label Number EM598711332US, addressed to the: Box Patent Application; Assistant Commissioner for Patents, Washington, D.C. 20231.

Valerie A. Milam

Valerie A. Milam

Signature of person mailing paper

WARNING: *Certificate of mailing (first class) or facsimile transmission procedures of 37 C.F.R. 1.8 cannot be used to obtain a date of mailing or transmission for this correspondence.*

****WARNING:*** *Each paper or fee filed by "Express Mail" **must** have the number of the "Express Mail" mailing label placed thereon prior to mailing. 37 C.F.R. 1.10(b).*

2. Benefit of Prior U.S. Application(s) (35 U.S.C. 119(e), 120, or 121)

NOTE: If the new application being transmitted is a continuation, CIP or divisional, of a parent case, or where the parent case is an International Application which designated the U.S., or the benefit of a prior **provisional** application is claimed, then check the following item and complete section as follows.

() The new application being transmitted claims the benefit of prior U.S. application(s).

2.1 Relate Back

WARNING: If an application claims the benefit of the filing date of an earlier filed application under 35 U.S.C. 120, 121 or 365(c), the 20-year term of that application will be based upon the filing date of the earliest U.S. application that the application makes reference to under 35 U.S.C. 120, 121 or 365(c). (35 U.S.C. 154(a)(2) does not take into account, for the determination of the patent term, any application on which priority is claimed under 35 U.S.C. 119, 365(a) or 365(b).) For a CIP application, applicant should review whether any claim in the patent that will issue is supported by an earlier application and, if not, the applicant should consider canceling the reference to the earlier filed application. The term of a patent is not based on a claim-by-claim approach. See Notice of April 14, 1995, 60 Fed Reg. 20,195, at 20,205.

(complete the following, if applicable)

Amend the specification by inserting, before the first line, the following sentence:

A. 35 U.S.C. 120, 121 and 365(c)

() "This is a
- ( ) continuation
- ( ) continuation-in-part
- ( ) divisional

of copending application(s) serial number filed on ."

( ) International Application_____ filed on_____ and which designated the U.S."

Note: The proper reference to a prior filed PCT application that entered the U.S. national phase is the U.S. serial number and the filing date of the PCT application that designated the U.S Moreover, (1) Where the application being transmitted adds subject matter to the International Application, then the filing can be as a continuation-in-part or (2) if it is desired to do so for other reasons then the filing can be as a continuation.

( ) "The nonprovisional application designated above, namely application no._____, filed_____, claims the benefit of U.S. Provisional Application(s) No(s).:

{list application no(s). and filing date(s)}

B. 35 U.S.C. 119(e) (Provisional Application)

( ) "This application claims the benefit of U.S. Provisional Application(s) No(s).:

{list application no(s). and filing date(s)}

2.2 Relate Back—35 U.S.C. 119 Priority Claim for Prior Application

The prior U.S. application(s), including any prior International Application designating the U.S., identified above in item 2.1(A), in turn itself claim(s) foreign priority(ies) as follows:

{list country, application no(s). and filing date(s)}

The certified copy(ies) has (have)

( ) been filed on_____, in prior application serial no.____, which was filed on___.

( ) is (are) attached.

2.3 Maintenance of Copendency of Prior Application

NOTE: The PTO finds it useful if a copy of the petition filed in the prior application extending the term for response is filed with the papers constituting the filing of the continuation application. Notice of November 5, 1985 (1060 O.G. 27).

A. ( ) Extension of time in prior application

*(This item **must** be completed and the papers filed **in the prior application** if the period set in the prior application has run.)*

( ) A petition, fee and response extends the term in the pending **prior** application until Extension of_____.

( ) A **copy** of the petition filed in prior application is attached.

B. ( ) Conditional Petition for Extension of Time in Prior Application

(complete this item, if previous item not applicable)

( ) A conditional petition for extension of time is being filed in the pending **prior** application.

( ) A **copy** of the conditional petition filed in the prior application is attached.

2.4 Further Inventorship Statement Where Benefit of Prior Application(s) Claimed

(complete applicable item A, B and/or C below)

A. ( ) This application discloses and claims only subject matter disclosed in the prior application whose particulars are set out above and the inventor(s) In this application are

( ) the same.

( ) less than those named in the prior application. It is requested that the following inventor(s) identified for the prior application be deleted:

{ type name(s) of inventor(s) to be deleted }

B. ( ) This application discloses and claims additional disclosure by amendment and a new declaration or oath is being filed. With respect to the prior application, the inventor(s) in this application are

( ) the same.

( ) the following additional inventor(s) have been added:

(type name(s) of inventor(s) to be added)

C. ( ) The inventorship for all the claims in this application are

( ) the same.

( ) not the same. An explanation, including the ownership of the various claims at the time the last claimed invention was made

( ) is submitted.

( ) will be submitted.

2.5 Abandonment of Prior Application *(if applicable)*

( ) Please abandon the prior application at a time while the prior application is pending, or when the petition for extension of time or to revive In that application is granted, and when this application is granted a filing date, so to make this application copending with said prior application.

NOTE: According to the Notice of May 13, 1983 (103, TMOG 6-7), the filing of a continuation or continuation-in-part application is a proper response with respect to a petition for extension of time or a petition to revive and should include the express abandonment of the prior application conditioned upon the granting of the petition and the granting of a filing date to the continuing application.

2.6 Petition for Suspension of Prosecution for the Time Necessary to File an Amendment

NOTE: Where it is possible that the claims on file will give rise to a first action final for this continuation application and for some reason an amendment cannot be filed promptly (e.g, experimental data is being gathered) it may be desirable to file a petition for suspension of prosecution for the time necessary.

(check the next Item, if applicable)

( ) There is provided herewith a Petition To Suspend Prosecution for the Time Necessary to File An Amendment (New Application Filed Concurrently)

2.7 Small Entity (37 CFR § 1.28(a))

( ) Applicant has established small entity status by the previous submission of a statement in prior application serial no.____ on ____.

( ) A copy of the statement previously filed is included.

WARNING: See 37 CFR § 1.28(a).

2.8. Notification in Parent Application of this Filing

( ) A notification of the filing of this
(check one of the following)

( ) continuation
( ) continuation-in-part
( ) divisional

is being filed in the parent application, from which this application claims priority under 35 U.S.C. § 120.

2.9 Incorporation by Reference

() the entire disclosure of the prior application, from which a copy of the oath or declaration is supplied, is considered to be part of the disclosure of the accompanying application and is hereby incorporated by reference therein.

3. Papers Enclosed Which are Required for Filing Date Under 37 CFR 1.53(b) (Regular) or 37 CFR 1.153 (Design) Application

(X) 13 Pages of specification
(X) 6 Pages of claims
(X) 1 Pages of Abstract
(X) 3 Sheets of drawing
(X) formal
() informal

4. Additional papers enclosed

() Amendment to claims:

( ) **Cancel** in this application claims_____ before calculating the filing fee. (At least one original independent claim must be retained for filing purposes).

( ) **Add** the claims shown in the attached amendment. (Claims added have been numbered consecutively following the highest numbered original claims).

( ) Preliminary Amendment
( ) Information Disclosure Statement (37 C.F.R. 1.98)
( ) Form PTO-1449
( ) Citations
( ) Declaration of Biological Deposit
( ) Special Comments
( ) Other

5. Declaration or oath (including power of attorney)

(X) ENCLOSED.
(X) Newly executed (original or copy)
() Copy from prior application No. 0 /_____ (37 CFR 1.63(d)- continuation/divisional)

( ) DELETION OF INVENTOR(S) - signed statement attached deleting inventor(s) named in the above-noted prior application (37 CFR 1.63(d) and 1.33(b))

Declaration or Oath executed by: (check **all** applicable boxes)
() inventor(s).
( ) legal representative of inventor(s). 37 CFR 1.42 or 1.43
( ) joint inventor or person showing a proprietary interest on behalf of inventor who refused to sign or cannot be reached.
( ) this is the petition required by 37 CFR 1.47 and the statement required by 37 CFR 1.47 is also attached. See item 13 below for fee.

() NOT ENCLOSED.
( ) Application is made by a person authorized under 37 CFR 1.41(c) on behalf of all the above named inventor(s). The declaration or oath, along with the surcharge required by 37 CFR 1.16(e) can be filed subsequently.
( ) Showing that the filing is authorized. (Not required unless called into question. 37 CFR 1.41(d)).

6. Inventorship Statement

WARNING: If the named inventors are each not the inventors of all the claims an explanation, including the ownership of the various claims at the time the last claimed invention was made, should be submitted.

The inventorship for all the claims in this application are:
() The same
or

() Not the same. An explanation, including the ownership of the various claims at the time the last claimed invention was made,

() is submitted

() will be submitted.

7. Language

(X) English

() Non-English

() the attached translation is a verified translation. 37 CFR 1.52(d).

8. Assignment

(X) An assignment of the invention to Cisco Technology, Inc.

(X) is attached. (A separate "ASSIGNMENT COVER LETTER ACCOMPANYING NEW PATENT APPLICATION" is also attached.)

() will follow.

() The prior application is assigned of record to __(copy attached).

9. Certified Copy - Foreign Priority Claim Under 35 U.S.C. 119

Certified copy(ies) of application(s)

{list country, application no(s). and filing date(s)}

from which priority is claimed

() is (are) attached.

() will follow.

NOTE: The foreign application forming the basis for the claim for priority **must** be referred to in the **oath** or **declaration**. 37 CFR 1.55(a) and 1.63.

NOTE: This item is for any foreign priority for which the application being filed directly relates. If any parent U.S. application or International Application form which this application claims benefit under 35 U S.C. 120 is itself entitled to priority from a prior foreign application then complete item 17 on the ADDED PAGES FOR NEW APPLICATION TRANSMITTAL WHERE BENEFIT OR PRIOR U.S. APPLICATION(S) CLAIMED.

10. Fee Calculation (37 C.F.R. 1.16)

A. (X) Regular Application

CLAIMS AS FILED				
	Number Filed	Number Extra	Rate	Basic Fee $690.00
Total Claims (37 CFR 1.16(c))	20 - 20 =	0	x $ 18.00	$ 0
Independent Claims (37 CFR 1.16(b))	7 - 3 =	4	x $ 78.00	$312
Multiple dependent claim(s), if any (37 CFR 1.16(d))	0	0	x $ 260.00	$ 0.00

( ) Amendment canceling extra claims enclosed.
( ) Amendment deleting multiple dependencies enclosed.
( ) Fee for extra claims is not being paid at this time.

NOTE: If the fees for extra claims are not paid on filing they must be paid or the claims canceled by amendment, prior to the expiration of the time period set for response by the Patent and Trademark Office in any notice of fee deficiency. 37 CFR 1.16(d).

Filing Fee Calculation $ 1,002.00

B. () Design Application
($330.00 - 37 CFR 1.16(f))

Filing Fee Calculation $

11. Small Entity Statement(s)

() Verified Statement(s) that this is a filing by a small entity under 37 CFR 1.9 and 1.27 is(are) attached.

Filing Fee Calculation (50% of **A** or **B** above) $

NOTE: Any excess of the full fee paid will be refunded if a verified statement and a refund request are filed within **2 months** of the date of timely payment of a full fee. 37 CFR 1.28(a).

12. Request for International-Type Search (37 C.F.R. 1.104(d))

( ) Please prepare an international-type search report for this application at the time when national examination on the merits takes place.

13. Fee Payment Being Made At This Time

( ) NOT ENCLOSED.

( ) No filing fee is to be paid at this time. (This and the surcharge required by 37 CFR 1.16(e) can be paid subsequently.)

(X) ENCLOSED

(X) Filing fee $ 1,002.00

(X) Recording assignment
($40.00; 37 CFR 1.21(h)(1)) $ 40.00

( ) petition fee for filing by other than all the inventors or person on behalf of the inventor where inventor refused to sign or cannot be reached. ($130.00; 37 CFR 1.47 & 1.17(h))
$____

( ) for processing an application with a specification is a non-English language. ($130.00 37 CFR 1.52(d) and 1.17(k))
$____

( ) processing and retention fee. ($130.00; 37 CFR 1.53(d) and 1.21(l))
$____

( ) Fee for international-type search report. ($40.00; 37 CFR 1.21(e))
$____

Total fees enclosed $ 1,042.00

14. Method of Payment of Fees

(X) Check in the amount of $ 1,042.00

() Charge Account No. 50-0902 in the amount of $ A duplicate of this transmittal is attached.

15. Authorization to Charge Additional Fees

WARNING: If no fees are to be paid on filing the following items should **not** be completed

WARNING: Accurately count claims, especially multiple dependent claims, to avoid unexpected high charges, if extra claim charges are authorized.

(X) The Commissioner is hereby authorized to charge the following additional fees by this paper and during the entire pendency of this application to Account No. 50-0902, **identifying our Attorney Docket No.** (72255/02662).

(X) 37 CFR 1.16(a), (f), or (g) (filing fees)
(X) 37 CFR 1.16(b), (c) and (d) (presentation of extra claims)
(X) 37 CFR 1.17 (application processing fees)

( ) 37 CFR 1.16(e) (surcharge for filing the basic filing fee and/or declaration on a date later than the filing date of the application)

( ) 37 CFR 1.17(a)(1)-(5) (extension fees pursuant to 37 CFR 1.136(a))

( ) 37 CFR 1.18 (issue fee at or before mailing Notice of Allowance, pursuant to 37 CFR 1.311(b))

16. Instruction As To Overpayment

() Credit Account No. 50-0902, **identifying our Attorney Docket No. _____**.
(X) Refund

17. Incorporation by reference of added pages

(X) The following pages are incorporated by reference:

(X) "Assignment Cover Letter Accompanying New Application"; number of pages added 3

() Added Pages For Papers Referred To In Item 4 Above; number of pages added

( ) Plus added pages deleting names of inventor(s) named in prior application(s) who is/are no longer inventor(s) of the subject matter claimed in this application; number of pages added_ _____.

(X) no further pages form a part of this Transmittal. The transmittal ends with this page.

Date: June 30, 2000

Michael A. Jaffe
Registration No. 36,326

ARTER & HADDEN LLP
1100 Huntington Building
925 Euclid Avenue
Cleveland, Ohio 44115-1475
Phone: (216) 696-3394
Fax: (216) 696-2645

DOCKET NO. 72255/02662

REPEAT FUNCTION FOR PROCESSING OF REPETITIVE INSTRUCTION STREAMS

Field of Invention

The present invention generally relates to a system for executing a series of processor instructions, and more specifically a repeat function for efficient processing of repetitive instruction streams.

Background of the Invention

It has been recognized that the performance of embedded central processing units (CPUs) has been impaired due to the need for off-chip memory storage devices. In this regard, embedded CPUs typically take the form of a single-chip processor having some peripheral components on the processor chip. Memory associated with the processor for storing instructions and data is typically located off the processor chip. Accordingly, both processor instructions and data must often be read across a common bus (e.g., "Von Neumann bus") from an off-chip memory storage device. This consumes the bulk of the critical path for items such as "instruction decode" and "memory-to-register" data transfers.

Moreover, off-chip memory storage devices add to the dollar cost of such system incorporating CPUs. Furthermore, off-chip memory storage devices consume valuable real-estate of printed circuit boards (PCBs) which is often at a premium in such arrangements as mini-PCI (Peripheral Component Interconnect) and PCMCIA (Personal Computer Memory Card International Association).

In addition, power consumption is also greatly increased due to performing off-chip memory accesses, which in turn adversely affects the battery life in a wireless application.

In view of the foregoing observations, it has been recognized that an ideal design for processing memory transfer instructions, should seek to: (a) minimize the number of off-chip memory instruction fetch accesses to improve data transfer speed and minimize power consumption, and (b) reduce the size of the off-chip memory storage device to minimize use of real-estate area and production costs. These objective are particularly important in the case of wireless applications which use embedded CPUs.

One way in which the prior art has attempted to address the foregoing problems is by providing an on-chip cache memory. One drawback to this approach is that it adds significant production costs to produce the embedded CPU (e.g., to produce an ASIC). Other drawbacks may include absence of locality-of-reference, coherence problems, and thrashing problems depending on the application.

Another prior art solution has included the use of dual busses, namely a separate instruction bus and data bus (i.e., "Harvard bus"). One drawback of this approach is that a dual bus system is too power hungry and expensive for many embedded CPU applications. Furthermore, the pins needed to provide dual busses are often not available.

Other prior art approaches include the use of a "looping" execution method and an "unrolling" execution method to increase data transfer speed. These and other approaches also have significant drawbacks, as will be discussed below.

Summary of the Invention

According to the present invention there is provided a method of operating a processor to repeatedly execute at least one associated instruction, the method including the steps of : (a) loading a register with a count value indicative of the number of times

the associated instruction is to be executed; (b) fetching and executing a REPEAT instruction indicating the at least one associated instruction to be repeatedly executed; (c) fetching the at least one associated instruction; and (d) executing the at least one associated instruction for as many times as indicated by the count value.

In accordance with another aspect of the present invention there is provided a method of operating a processor to repeatedly execute one or more instructions, the method including the steps of : (a) fetching a REPEAT instruction; (b)executing a REPEAT instruction, wherein execution of the REPEAT instruction stores in a register a count value indicative of the number of times one or more associated instructions are be executed; (c) fetching the one or more associated instructions; and (d) executing the associated instruction for as many times as indicated by the count value.

In accordance with still another aspect of the present invention there is provided a method of operating a processor to repeatedly execute one or more instructions, the method including the steps of: (a) loading a register with a count value indicative of the number of times one or more associated instructions are to be executed; (b) fetching and executing a REPEAT instruction indicating the one or more associated instructions that are to be repeatedly executed; (c) incrementing a program counter; (d) fetching the one or more associated instructions; and (e) executing the one or more associated instruction for as many times as indicated by a count value stored in a count register.

In accordance with yet another aspect of the present invention there is provided a processor for repeatedly execute at least one associated instruction, said processor comprising: load means for loading a register with a count value indicative of the number of times the associated instruction is to be executed; first fetch means for a REPEAT instruction indicating the at least one associated instruction to be repeatedly executed; first execute means for executing the REPEAT instruction indicating the at least one associated instruction to be repeatedly executed; second fetch means for

fetching the at least one associated instruction; and first execute means for executing the at least one associated instruction for as many times as indicated by the count value.

In accordance with yet another aspect of the present invention there is provided a processor for repeatedly executing one or more instructions, comprising: first fetch means for fetching a REPEAT instruction; first execute means for executing a REPEAT instruction, wherein execution of the REPEAT instruction stores in a register a count value indicative of the number of times one or more associated instructions are be executed; second fetch means for fetching the one or more associated instructions; and second execute means for executing the associated instruction for as many times as indicated by the count value.

In accordance with yet another aspect of the present invention there is provided a processor for repeatedly executing one or more instructions, comprising: load means for loading a register with a count value indicative of the number of times one or more associated instructions are to be executed; first fetch means for fetching a REPEAT instruction indicating the one or more associated instructions that are to be repeatedly executed; first execute means for executing the REPEAT instruction indicating the one or more associated instructions that are to be repeatedly executed; means for incrementing a program counter; second fetch means for fetching the one or more associated instructions; and second execute means for executing the one or more associated instruction for as many times as indicated by a count value stored in a count register.

In accordance with still another aspect of the present invention there is provided a processor for repeatedly executing one or more processor instructions, said processor comprising: a memory address register associated with a main memory; a memory data register associated with the main memory; a memory control for generating memory control signals; a program counter for storing a memory address location of the main memory where an instruction is to be fetched; an instruction register for storing an instruction that is to be executed; at least one general purpose register; decode and

execute control logic for decoding and executing an instruction stored in the instruction register; and a state machine for controlling the fetching and repeated execution one or more associated instructions.

An advantage of the present invention is the provision of a repeat function which minimizes the number of off-chip memory instruction fetch accesses.

Another advantage of the present invention is the provision of a repeat function which conserves power in an embedded processor system, thus maximizing battery life.

Another advantage of the present invention is the provision of a repeat function which minimizes the size of the off-chip memory storage device, thus conserving real estate.

Still another advantage of the present invention is the provision of a repeat function which minimizes production costs of embedded processor systems.

Yet another advantage of the present invention is the provision of a repeat function which allows the use of a Von Neumann bus configuration, but provides the effect of a Harvard bus configuration.

Yet another advantage of the present invention is the provision of a repeat...until function that allows for repetition of a group of instructions without an instruction fetch.

Still other advantages of the invention will become apparent to those skilled in the art upon a reading and understanding of the following detailed description, accompanying drawings and appended claims.

Brief Description of the Drawings

The invention may take physical form in certain parts and arrangements of parts, a preferred embodiment and method of which will be described in detail in this

specification and illustrated in the accompanying drawings which form a part hereof, and wherein:

Fig. 1A illustrates exemplary code for the prior art looping execution method;

Fig. 1B illustrates exemplary code for the prior art unrolling execution method; and

Fig. 2 illustrates exemplary code for a repeat function according to a preferred embodiment of the present invention;

Fig. 3 illustrates a state machine for providing the repeat function according to a preferred embodiment of the present invention;

Fig. 4 illustrates the operation of executing a REPEAT instruction requiring five repeats of an associated instruction, in accordance with the state machine shown in Fig. 1;

Fig. 5 shows a timing diagram corresponding to the operation shown in Fig. 4; and

Fig. 6 is a block diagram showing the basic elements of a processor for executing the REPEAT instruction, according to a preferred embodiment of the present invention.

Detailed Description of the Preferred Embodiment

In computer program execution, a fetch operation transfers the contents of a specific main memory location to the processor or central processing unit (CPU). To start a fetch operation, the CPU must send the address of the desired memory location to the main memory. The CPU contains a register known as the program counter (PC) that contains the address in the main memory of the instruction to be executed. Execution of a given instruction is generally comprised of a two-phase procedure. In the first phase, called "instruction fetch," the instruction is fetched from the main memory location

whose address is in the PC. This instruction is placed in the instruction register (IR) in the CPU. At the start of the second phase, called "instruction execute," the operation field of the instruction in the IR is examined to determine which operation is to be performed. The specified operation is then performed by the CPU. In the case where a shared bus is used for transfer of both instructions and data stored in the main memory (i.e., "Von Neumann bus"), the "fetching" and "execution" of a memory transfer instruction (e.g., memory read (MRD) and memory write instructions), will require use of the same bus.

Figs. 1A and 1B show prior art methods for executing a series of repetitive instructions, such as memory transfer instructions (MRD - "memory read"). Fig. 1A illustrates a "looping" method, while Fig. 1B illustrates an "unrolling" method. As will be readily understood by those skilled in the art, the looping method requires one instruction fetch operation each time the MRD instruction is encountered, and one instruction fetch operation each time the DBcc ("decrement and branch") instruction is encountered. Thus, each 3 consecutive clock cycles of a single loop consist of: (1) fetch MRD, (2) perform memory data transfer (i.e. execute memory read instruction), and (3) fetch DBcc (decrement and branch). Accordingly, the "looping" method provides an effective data transfer rate of 1 data transfer for every 3 clock cycles on a shared instruction/data bus.

The prior art "unrolling" method shown in Fig. 1B, is somewhat better than the "looping" method. As noted above, one instruction fetch operation is executed each time the MRD ("memory read") instruction is encountered. Thus, each two consecutive clock cycle consists of: (1) fetch MRD, and (2) perform memory data transfer (i.e., execute memory read instruction). Accordingly, the "unrolling" method provides an effective data transfer rate of 1 data transfer for every 2 clock cycles on a shared instruction/data bus. However, it should be understood that the "unrolling"

method requires additional memory space for storing the plurality of MRD instructions. Consequently, a larger memory storage device is needed which adds to production cost.

Fig. 2 illustrates a special instruction referred to herein as a REPEAT instruction for repetitively executing an associated instruction. The number of repetitions is specified by a COUNT value preloaded into a register (R0) which said REPEAT instruction is associated with, or alternatively, specified as part of the REPEAT instruction (REPEAT N), where N specifies the number of repetitions. As will be described below, the REPEAT instruction eliminates the need to repetitively fetch on the same bus as a memory transfer. This allows for the same effect as a Harvard bus cycle, but without the added cost of provided separate instruction and data buses.

Referring now to Fig. 3, there is shown a state machine 100 for decoding a REPEAT instruction, according to a preferred embodiment of the present invention. In STATE 1 (IDLE) a COUNT value has been preloaded into a register R0 indicative of the number of times an instruction (referred to herein as $INST_R$) is to be repeated upon encountering a REPEAT instruction. Upon transition from STATE 1 to STATE 2 (FIRST FETCH), a REPEAT instruction is fetched.

In STATE 2, "decrement COUNT" and "re-execute" signals are asserted. Assertion of the "re-execute" signal results in the fetching and execution of the instruction $INST_R$, upon transition from STATE 2 to STATE 3 (RE-EXECUTE). $INST_R$ is the next consecutive instruction following the REPEAT instruction (i.e., the instruction associated with the REPEAT instruction).

In STATE 3, a "re-execute" signal is asserted and if COUNT is greater than zero, and $INST_R$ is again executed. A "decrement count" signal is also asserted to decrement the COUNT by one, each time $INST_R$ is executed. Once the COUNT is less than or equal to zero, an "increment PC" signal is asserted, and there is a transition back to STATE 1. Upon transition to STATE 1, the address stored in the program counter (PC) is incremented.

Referring to Fig. 4, there is shown a table illustrating the a REPEAT instruction for repeating an instruction $INST_R$ five (5) times. "Program counter" refers to the address (hex) stored in the program counter, the "transitions" refer to locations in the state diagram shown in Fig. 3, while the "operation" indicates what type of operation is occurring. A corresponding timing diagram is shown in Fig. 5. As can be seen in Figs. 4 and 5, the program counter remains unchanged as the same instruction $INST_R$ is repeatedly executed. Once the instruction $INST_R$ has been executed the number of times specified by COUNT, the program counter is incremented.

It should be appreciated that the instruction that is repeated (i.e., $INST_R$) may be any type of instruction, including, but not limited to, memory data transfer instructions, such as memory read (MRD) or memory write instructions, and shift instructions.

In effect, the REPEAT instruction acts as an instruction cache which holds one repeatedly executed instruction (i.e., $INST_R$). Accordingly, the present invention provides the benefits of an on-chip instruction cache, without the expense and problems associated with implementing an instruction cache.

Referring now to Fig. 6, an exemplary processor for implementing the REPEAT instruction is illustrated. It should be appreciated that the processor illustrated in Fig. 6 is provided solely for the purposes of illustrating a preferred embodiment of the present invention, and that other processor designs (including non-RISC processors) may also be used for implementation of the REPEAT instruction of the present invention.

The processor is generally comprised of a memory address register (MAR) 20, a memory data register (MDR) 30, a memory control 40, a program counter (PC) 50, a plurality of registers 60, an instruction register (IR) 70, an instruction buffer 80, an instruction decode and execute control logic 90, an arithmetic logic unit (ALU) 95, and a repeat state machine 100. The processor is connected with a main memory 10 for

exchange of data. It should be understood that not all interconnections among the processor elements are shown.

MAR 20 is used to hold the address of the location to or from which data is to be transferred. MDR 30 contains the data to be written into or read out of the addressed location. IR 70 contains the instruction that is being executed. Its output is available to the IR decode and execute control logic 90 that are needed to execute the instruction. PC 50 is a register that keeps track of the execution of a program. It contains the memory address of the instruction currently being executed. A plurality of general purpose registers 60 store various values needed during processing, such as the COUNT value associated with the REPEAT instruction. Programs typically reside in main memory 10 which interfaces with the processor via a bus.

In accordance with a preferred embodiment of the present invention, the processor is a RISC machine. Processor control is hard coded in a preferred embodiment, rather than software microcode. The following register transfer logic (RTL) is implemented for the REPEAT instruction:

IDLE:

IR ← Fetch [REPEAT instruction]
PC ← PC + 1
R0 ← R0
FIRST FETCH ← IDLE (transition A)

FIRST FETCH:

IR ← [$INST_R$]
PC ← PC
R0 ← R0 - 1
RE-EXECUTE ← FIRST FETCH (transition B)

RE-EXECUTE:

```
IR ← IR
R0 ← R0 - 1
IF (R0 > 0) THEN
        PC ← PC; RE-EXECUTE ← RE-EXECUTE (transition C)
ELSE
        PC ← PC + 1
        IDLE ← RE-EXECUTE (transition D)
```

As can be readily appreciated, the REPEAT instruction of the present invention is a significant improvement over the prior art. In this regard, after loading the COUNT and executing REPEAT instruction, one data transfer may be performed every clock cycle on a shared instruction/data bus, where $INST_R$ is a memory read/write instruction. Thus, it can be clearly observed that the present invention is 3 times as efficient as the prior art "looping" method, and is twice as efficient as the prior art "unrolling" method. Moreover, the REPEAT instruction of the present invention provides the added benefit of small code store. In addition, power consumption is also less due to reduced memory traffic. Accordingly, the present invention provides the same effect as a cache memory, but without the drawbacks inherent with use of a cache memory.

It should be further appreciated that the REPEAT instruction of the present invention does not need an instruction fetch, as required in the case of a "loop caching" method. In this regard, the program counter is effectively stalled on the same instruction which gets executed over and over again. With "loop caching" an instruction cache fetch must still be performed. "Loop caching" refers to an instruction caching variation which seeks to prioritize the caching of instruction streams which loop. Since a fetch instruction phase is required for "loop caching," performance will suffer. Moreover, there is the drawback with caching in regards to hardware overhead, and penalty for cache miss which is further elaborated below with regards to context switching.

In a further embodiment of the present invention, the REPEAT instruction takes the form of a REPEAT...UNTIL instruction using a multiple instruction buffer. In this regard, multiple instructions are repeatedly executed. The REPEAT...UNTIL instruction achieves the same advantage provided by "loop caching" so that the number of off-chip fetch operations are reduced. The key difference is that the REPEAT...UNTIL instruction utilizes an instruction buffer 80 (Fig. 6), rather than an instruction cache. Thus, as with the REPEAT instruction, multiple instruction fetches are not required. The contents of instruction buffer 80 are loaded from the operations nested between the REPEAT and the UNTIL program statements.

It should be understood that with context switching, the "loop caching" method described above will suffer. In this regard, the loop can become un-cached when the next context executes, since the locality of reference is lost. Therefore, when the suspended code thread is resumed, the loop performance may suffer. In the presence of very rapid context switching ("thrashing") loop instruction caches can thus become useless. However, the REPEAT instruction and the REPEAT...UNTIL instruction do not have this drawback, since the contents of the instruction buffer are locked in and are preserved while the context is pre-empted. Thus, resumption of a switched context that was using a REPEAT or REPEAT..UNTIL instruction does not suffer in performance. In should be appreciated that, in effect, the REPEAT instruction has an instruction buffer containing one instruction.

The utility of using the REPEAT instruction to execute a single instruction over and over again, should be fully appreciated. In the case of code threads that rapidly context switch in a pre-emptive multi-tasking environment, repetition of instructions is of particular importance. For example, a MRD (memory read) instruction can be used to target a FIFO memory, which needs servicing by a code thread. The instruction contains a side effect which suspends the active context until it is interrupted (e.g., the FIFO needs

servicing again). Hence, the instruction loop includes one instruction which needs to be continuously executed until the desired amount of data has been sent to the FIFO. For example, the following program code sample will read with post increment (address contents of register R0) from memory to FIFO 100 times, and suspend that context until next time of service after each read is performed.

```
REPEAT 100
MRD [R0]+, FIFO, WAIT
```

An example of REPEAT..UNTIL is as follows:

```
REPEAT 100
        MRD [R1]+, R2
        MRD [R3]+, R4
        ADD  R7, R2, R4
        MWR [R5]+, R7
UNTIL
```

This loop reads data into registers R2 and R4 using post increment address contents of register R1 and R3. The result is added and placed into register R7, where it is stored to memory at the R5 address, post increment. This is done 100 times in a row per the REPEAT N instruction (where N = 100).

The present invention has been described with reference to a preferred embodiment. Obviously, modifications and alterations will occur to others upon a reading and understanding of this specification. It is intended that all such modifications and alterations be included insofar as they come within the scope of the appended claims or the equivalents thereof.

Having thus described the invention, it is now claimed:

1. A method of operating a processor to repeatedly execute at least one associated instruction, comprising:

loading a register with a count value indicative of the number of times the associated instruction is to be executed;

fetching and executing a REPEAT instruction indicating the at least one associated instruction to be repeatedly executed;

fetching the at least one associated instruction; and

executing the at least one associated instruction for as many times as indicated by the count value.

2. A method of operating a processor to repeatedly execute one or more instructions, comprising:

fetching a REPEAT instruction;

executing a REPEAT instruction, wherein execution of the REPEAT instruction stores in a register a count value indicative of the number of times one or more associated instructions are be executed;

fetching the one or more associated instructions; and

executing the associated instruction for as many times as indicated by the count value.

3. A method of operating a processor to repeatedly execute one or more instructions, comprising:

loading a register with a count value indicative of the number of times one or more associated instructions are to be executed;

fetching and executing a REPEAT instruction indicating the one or more associated instructions that are to be repeatedly executed;

incrementing a program counter;

fetching the one or more associated instructions; and

executing the one or more associated instruction for as many times as indicated by a count value stored in a count register.

4. A method of operating a processor according to claim 3, wherein said count value is stored in said count register before execution of said REPEAT instruction.

5. A method of operating a processor according to claim 3, wherein said REPEAT instruction includes the count value that is stored in said count register, wherein execution of the REPEAT instruction stores the count value in said count register.

6. A method of operating a processor according to claim 3, wherein said method further comprises:

incrementing the program counter after the one or more associated instructions have been executed for as many times as indicated by the count value.

7. A method according to claim 3, wherein method further comprises:

decrementing said count value stored in said register each time said one or more associated instructions are executed; and

determining whether said count value is less than or equal to zero.

8. A processor for repeatedly execute at least one associated instruction, said processor comprising:

load means for loading a register with a count value indicative of the number of times the associated instruction is to be executed;

first fetch means for a REPEAT instruction indicating the at least one associated instruction to be repeatedly executed;

first execute means for executing the REPEAT instruction indicating the at least one associated instruction to be repeatedly executed;

second fetch means for fetching the at least one associated instruction; and

first execute means for executing the at least one associated instruction for as many times as indicated by the count value.

9. A processor for repeatedly executing one or more instructions, comprising:

first fetch means for fetching a REPEAT instruction;

first execute means for executing a REPEAT instruction, wherein execution of the REPEAT instruction stores in a register a count value indicative of the number of times one or more associated instructions are be executed;

second fetch means for fetching the one or more associated instructions; and

second execute means for executing the associated instruction for as many times as indicated by the count value.

10. A processor for repeatedly executing one or more instructions, comprising:

load means for loading a register with a count value indicative of the number of times one or more associated instructions are to be executed;

first fetch means for fetching a REPEAT instruction indicating the one or more associated instructions that are to be repeatedly executed;

first execute means for executing the REPEAT instruction indicating the one or more associated instructions that are to be repeatedly executed;

means for incrementing a program counter;

second fetch means for fetching the one or more associated instructions; and

second execute means for executing the one or more associated instruction for as many times as indicated by a count value stored in a count register.

11. A processor according to claim 10, wherein said count value is stored in said count register before execution of said REPEAT instruction.

12. A processor according to claim 10, wherein said REPEAT instruction includes the count value that is stored in said count register, wherein execution of the REPEAT instruction stores the count value in said count register.

13. A processor according to claim 10, wherein said processor further comprises:

means for incrementing the program counter after the one or more associated instructions have been executed for as many times as indicated by the count value.

14. A processor according to claim 10, wherein processor further comprises:

means for decrementing said count value stored in said register each time said one or more associated instructions are executed; and

means for determining whether said count value is less than or equal to zero.

15. A processor for repeatedly executing one or more processor instructions, said processor comprising:

a memory address register associated with a main memory;

a memory data register associated with the main memory;

a memory control for generating memory control signals;

a program counter for storing a memory address location of the main memory where an instruction is to be fetched;

an instruction register for storing an instruction that is to be executed;

at least one general purpose register;

decode and execute control logic for decoding and executing an instruction stored in the instruction register; and

a state machine for controlling the fetching and repeated execution of one or more associated instructions.

16. A processor according to claim 15, wherein said processor further comprises an instruction buffer for storing the one or more associated instructions.

17. A processor according to claim 15, wherein said general purpose register includes a first register for storing a count value indicative of the number of times the one or more associated instructions are to be repeatedly executed.

18. A processor according to claim 17, wherein said state machine generates signals for decrementing the count value stored in the first register.

19. A processor according to claim 17, wherein said state machine generates a signal for executing an instruction stored in said instruction register.

20. A processor according to claim 17, wherein said state machine generate a signal for incrementing said program counter.

ABSTRACT

A REPEAT instruction for repeated execution of an associated instruction ($INST_R$). Once a program counter stores the address for the instruction to be repeated, it remains unchanged until the associated instruction ($INST_R$) has been executed the number of times indicated by a COUNT value in a preloaded register, or alternatively, by the REPEAT instruction itself. In this manner, the present invention reduces the number of instruction fetches required to repeatedly execute the associated instruction ($INST_R$). Consequently, there is a significant improvement in the efficiency of the program code execution.

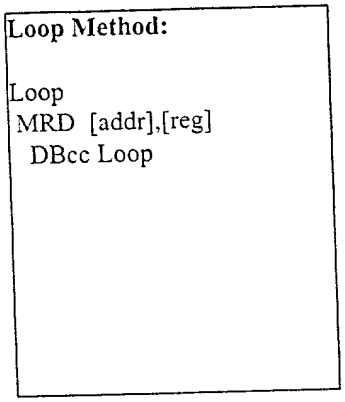

Fig. 1A

PRIOR ART

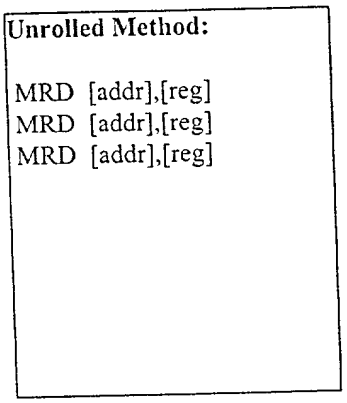

Fig. 1B

PRIOR ART

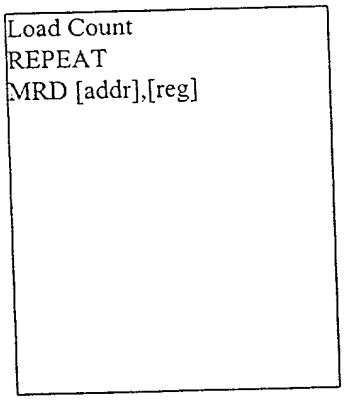

Fig. 2

100

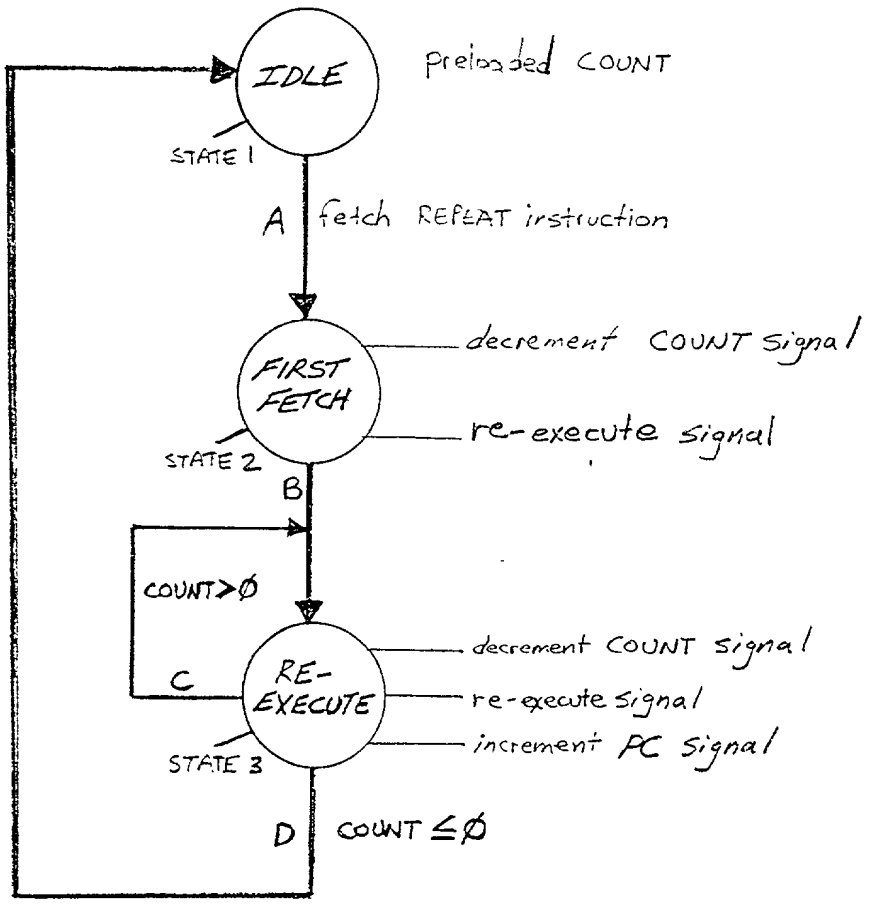

Fig. 3

Fig. 4

PROGRAM COUNTER(PC)	TRANSITION	OPERATION
x 10	A	fetch REPEAT instruction
x 11	B	fetch instruction to be repeated ($INST_R$)
x 11	C	execute $INST_R$
x 11	C	execute $INST_R$
x 11	C	execute $INST_R$
x 11	C	execute $INST_R$
x 11	C	execute $INST_R$
x 12	D	fetch next instruction

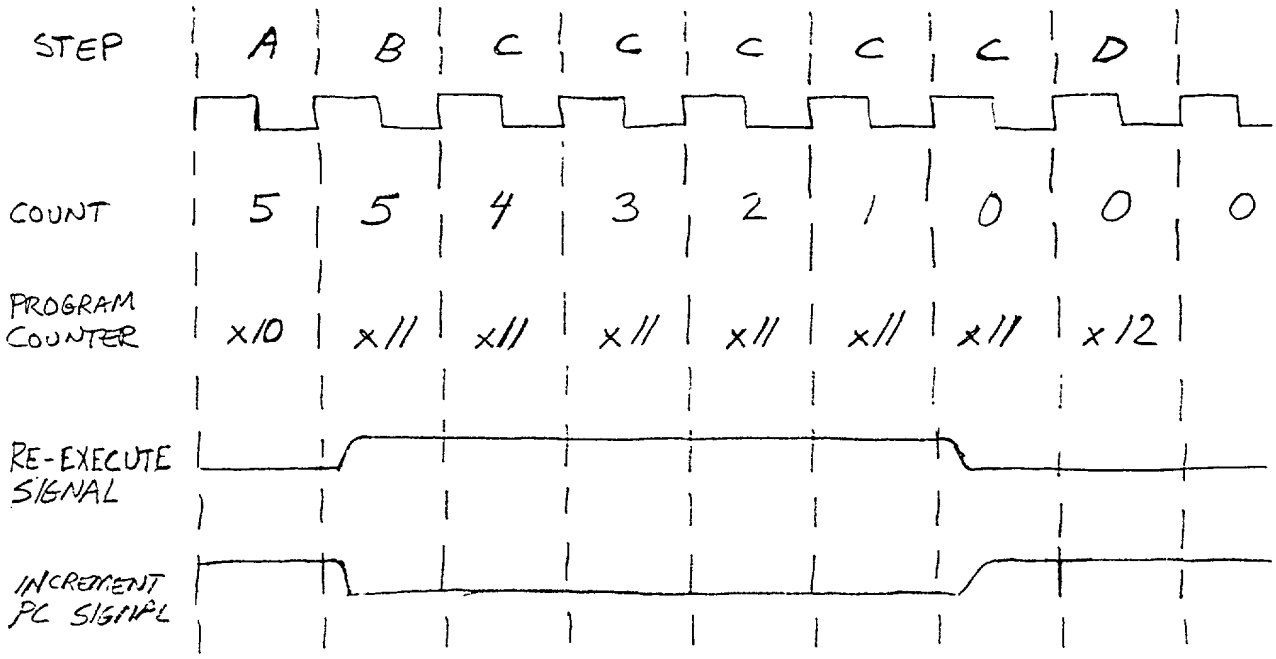
STEP
A
B
C
C
C
C
C
D
COUNT
5
5
4
3
2
1
0
0
0
PROGRAM COUNTER
x10
x11
x11
x11
x11
x11
x11
x12
RE-EXECUTE SIGNAL
INCREMENT PC SIGNAL

Fig. 5

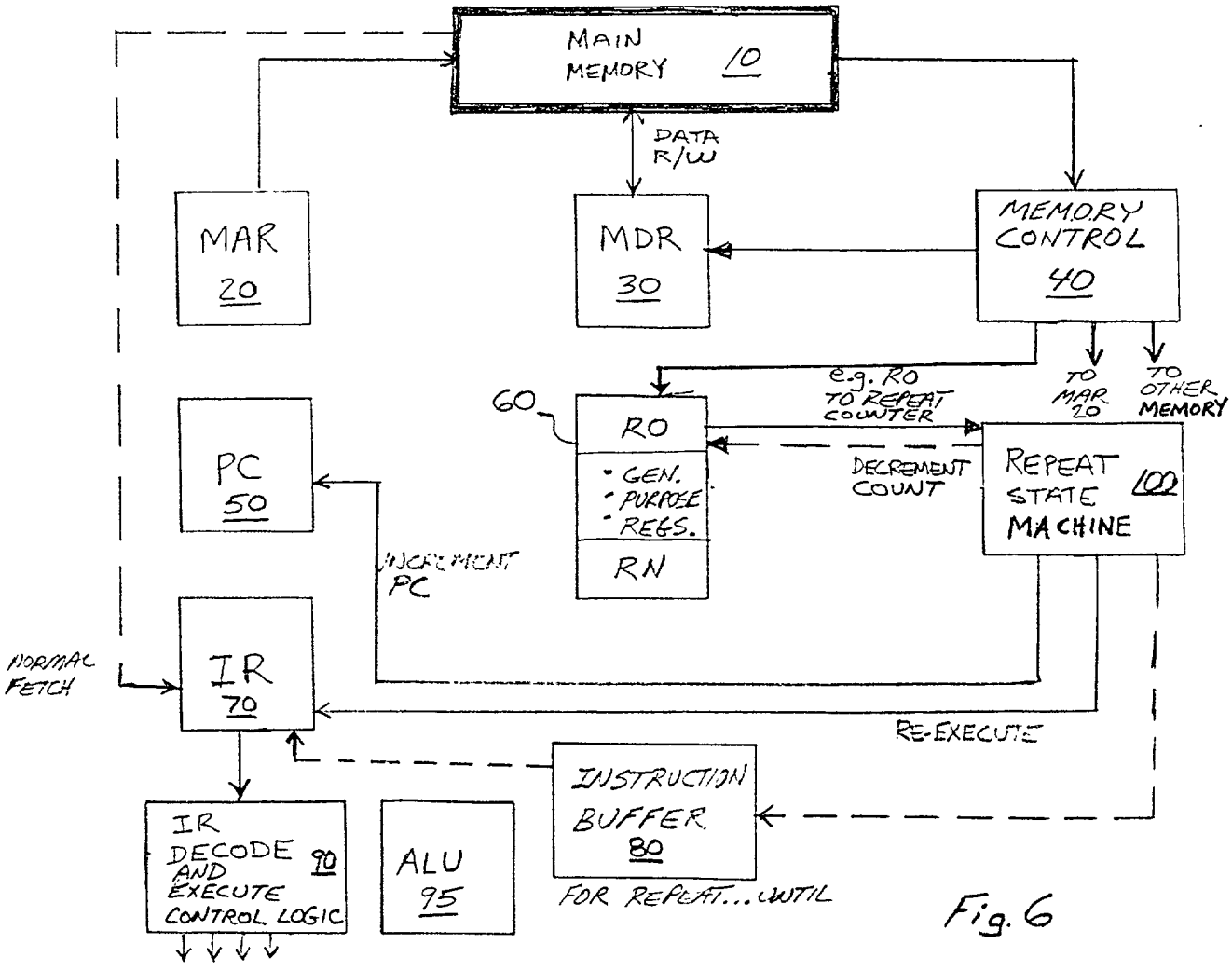
MAIN MEMORY 10
DATA R/W
MAR 20
MDR 30
MEMORY CONTROL 40
TO MAR 20
TO OTHER MEMORY
e.g. R0 TO REPEAT COUNTER
60
R0
• GEN. • PURPOSE • REGS.
RN
DECREMENT COUNT
REPEAT STATE MACHINE 100
PC 50
INCREMENT PC
NORMAL FETCH
IR 70
RE-EXECUTE
INSTRUCTION BUFFER 80
IR DECODE AND EXECUTE CONTROL LOGIC 90
ALU 95
FOR REPEAT... UNTIL

Fig. 6

Docket No. 72255/02662

DECLARATION FOR PATENT APPLICATION

As a below named inventor, I hereby declare that:

My residence, post office address, and citizenship are as stated below next to my name.

I believe that I am the original, first and sole inventor (if only one name is listed below), or an original, first and joint inventor (if plural names are listed below) of the subject matter which is claimed, and for which a patent is sought on the invention entitled:

REPEAT FUNCTION FOR PROCESSING OF REPETITIVE INSTRUCTION STREAMS

the specification of which is attached hereto, unless the following box is checked:

__ was filed on June 30, 2000, as United States Application Number or PCT International Application Number _____.

I hereby state that I have reviewed and understand the contents of the above-identified specification, including the claims, as amended by any amendment referred to above.

I acknowledge the duty to disclose information which is material to patentability as defined in Title 37, Code of Federal Regulations, §1.56.

I hereby claim foreign priority benefits under Title 35, United States Code, §§119(a) - (d) or §365(b) of any foreign application(s) for patent or inventor's certificate, or §365(a) of any PCT International application which designated at least one country other than the United States, listed below and have also identified below any foreign application for patent or inventor's certificate, or PCT International application having a filing date before that of the application on which priority is claimed:

NONE

I hereby claim the benefit under Title 35, United States Code, §119(e) of any United States provisional application(s) listed below:

NONE

I hereby claim the benefit under Title 35, United States Code, §120 of any United States application(s), or §365(c) of any PCT International application designating the United States, listed below and, insofar as the subject matter of each of the claims of this application is not disclosed in the prior United States or PCT International application in the manner provided by

the first paragraph of Title 35, United States Code, §112, I acknowledge the duty to disclose information which is material to patentability as defined in Title 37, Code of Federal Regulations, §1.56 which became available between the filing date of the prior application and the national or PCT International filing date of this application:

NONE

I hereby appoint the following registered attorney(s) and/or agent(s) to prosecute this application and to transact all business in the Patent and Trademark Office connected therewith:

Alan J. Ross, Reg. No. 33,767
John X. Garred, Reg. No. 31,830
Michael A. Jaffe, Reg. No. 36,326
Susan L. Mizer, Reg. No. 38,245
Todd R. Tucker, Reg. No. 40,850
James C. Scott, Reg. No. 35,351
Jay P. Ryan, Reg. No. 37,064

Direct all telephone calls to : John X. Garred
at telephone number : (216) 696-3340

Direct all correspondence to : John X. Garred

ARTER & HADDEN LLP
1100 Huntington Building
925 Euclid Avenue
Cleveland, Ohio 44115-1475

I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under 18 U.S.C. 1001 and that such willful false statements may jeopardize the validity of the application or any patent issued thereon.

Full name of sole or first inventor: Kenneth W. Batcher

Inventor's signature : Kenneth W. Batcher

Date : Jun 30 2000

Residence : Hudson, Ohio
Citizenship : United States of America
Post Office Address : 1769 Mayflower Lane, Hudson, Ohio 44236

517001.1